Universitatea Naţională de Ştiinţă şi Tehnologie Politehnica Bucureşti
Facultatea de Electronică, Telecomunicaţii şi
Tehnologia Informaţiei

FIŞA DISCIPLINEI

1. Date despre program

1.1 Instituţia de învăţământ superior	**Universitatea Naţională de Ştiinţă şi Tehnologie Politehnica Bucureşti**
1.2 Facultatea	**Electronică, Telecomunicaţii şi Tehnologia Informaţiei**
1.3 Departamentul	**Dispozitive, Circuite şi Arhitecturi Electronice**
1.4 Domeniul de studii	Inginerie Electronică, Telecomunicaţii şi Tehnologii Informaţionale
1.5 Ciclul de studii	Masterat
1.6 Specializarea	Calcul Avansat în Sisteme Embedded

2. Date despre disciplină

2.1 Denumirea disciplinei (ro) (en)				Dezvoltare software şi testare Software Development Process and Testing			
2.2 Titularul activităţilor de curs				Conf. Dr. Radu Hobincu			
2.3 Titularul activităţilor de seminar / laborator				Conf. Dr. Radu Hobincu			
2.4 Anul de studiu	1	2.5 Semestrul	2	2.6. Tipul de evaluare	V	2.7 Regimul disciplinei	Ob
2.8 Tipul disciplinei		S	2.9 Codul disciplinei	2		2.10 Tipul de notare	Nota

3. Timpul total estimat (ore pe semestru al activităţilor didactice)

3.1 Număr de ore pe săptămână	4	Din care: 3.2 curs	2	3.3 seminar/laborator	2
3.4 Total ore din planul de învăţământ	56	Din care: 3.5 curs	28	3.6 seminar/laborator	28
Distribuţia fondului de timp:					ore
Studiul după manual, suport de curs, bibliografie şi notiţe Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate Pregătire seminarii/ laboratoare/proiecte, teme, referate, portofolii şi eseuri					75
Tutorat					5
Examinări					0
Alte activităţi (dacă există):					14

3.7 Total ore studiu individual	94.00
3.8 Total ore pe semestru	150
3.9 Numărul de credite	6

4. Precondiţii (acolo unde este cazul)

4.1 de curriculum	Cunoştinţe de programare în limbajele C/C++, Structuri de date şi algoritmi
4.2 de rezultate ale învăţării	Abilităţi de utilizare a calculatorului, competenţe tehnice.

5. Condiţii necesare pentru desfăşurarea optimă a activităţilor didactice (acolo unde este cazul)

5.1 Curs	Proiector şi acces la Internet.
5.2 Seminar/ Laborator/Proiect	Laborator cu acces la Internet pentru studenţi şi proiector.

6. Obiectiv general *(Se referă la intenţiile profesorilor pentru studenţi, la ceea ce studenţii vor fi învăţaţi în timpul cursului. Oferă o orientare cu privire la locul cursului în cadrul domeniului ştiinţific abordat, precum şi la rolul pe care acesta îl are în cadrul specializării studiate. Vor fi descrise de o manieră generală tematicile abordate, justificarea includerii cursului în planul de învăţământ al specializării studiate etc.)*

Obiectivul general este acela de a obișnui studentul cu procesul complet de dezvoltare software, care nu se rezumă doar la scrierea codului, cât și la toate procese conexe: compilare, testare, documentare și implementare. Astfel, studenții vor studia cum se utilizează eficient sistemele de control al versiunii, ce opțiuni au în privința instrumentelor eficiente de compilare, cum se testează aplicațiile folosind teste unitare, cum se generează documentație pentru o aplicație, și cum se folosesc sistemele de containere pentru integrare continuă, cros-compilare și implementare.

7. Competenţe *(Capacitatea dovedită de a utiliza cunoştinţe, aptitudini şi abilităţi personale, sociale şi/sau metodologice în situaţii de muncă sau de studiu şi pentru dezvoltarea profesională şi personală. Reflectă cerinţele angajatorilor.)*

Specifice	C3. Aplicarea cunoştinţelor, conceptelor şi metodelor de bază privitoare la arhitectura sistemelor de calcul, microprocesoare, microcontrolere, limbaje şi tehnici de programare (3/4)
Transversale (generale)	-

8. Rezultatele învăţării *(Sunt enunţuri sintetice referitoare la ceea ce un student va fi capabil să facă sau să demonstreze la finalizarea unui curs. Rezultatele învăţării reflectă realizările studentului şi mai puţin intenţiile profesorului. Rezultatele învăţării informează studenţii despre ceea ce se aşteaptă de la ei din punct de vedere al performanţei, pentru a obţine notele şi creditele dorite. Sunt definite în termeni concreţi, folosind verbe similare exemplelor de mai jos şi indică ceea ce se va urmări prin evaluare. Rezultatele învăţării vor fi astfel redactate încât să fie evidenţiată clar relaţia faţă de competenţele definite la punctul 7.)*

Cunoştinţe	*Rezultatul asimilării de informaţii prin învăţare. Cunoştinţele reprezintă ansamblul de fapte, principii, teorii şi practici legate de un anumit domeniu de muncă sau de studiu. Pot fi teoretice şi/sau faptice.* În urma acestui curs, studenții vor înțelege tot lanțul de instrumente necesare pentru aducerea unui software de la concepție la implementare și vor fi capabili să discearnă unde acesta nu funcționează și trebuie îmbunătățit.
Aptitudini	*Capacitatea de a aplica cunoştinţe şi de a utiliza know-how pentru a duce la îndeplinire sarcini şi a rezolva probleme. Aptitudinile sunt descrise ca fiind cognitive (implicând utilizarea gândirii logice, intuitive şi creative) sau practice (implicând dexteritate manuală şi utilizarea de metode, materiale, unelte şi instrumente).* Concret, studenții vor ști să folosească sistemul de control al versiunii Git cât și platforma Gitlab în scenarii complexe, vor putea să scrie propriile teste unitare folosind biblioteca Google Test, vor putea crea medii complexe de compilare eficiente folosind GNU Make sau CMake, vor putea genera documentație pentru codul scris folosind Doxygen și vor putea integra toate aceste instrumente în sisteme de integrare continuă folosind sistemul de containere Docker.

Responsabilitate şi autonomie	*Capacitatea cursantului de a aplica în mod autonom şi responsabil cunoştinţele şi aptitudinile sale.* În urma acestui curs, studenţii vor putea dezvolta în mod autonom, tot lanţul de dezvoltare software, plecând de la scrierea codului, până la implementare în producţie.

9. Metode de predare *(Se vor avea în vedere metode care să asigure predarea centrată pe student. Se va descrie modul în care se asigură participarea studenţilor la stabilirea propriului parcurs de învăţare, cum se identifică eventualele rămâneri în urmă şi ce măsuri remediale se adoptă în astfel de cazuri.)*

Predarea se va face într-un mod interactiv, prin exemple practice, prezentate în timp real, legate de proiecte software complexe.

10. Conţinuturi

CURS

Capitolul	Conţinutul	Nr. ore
1	Sisteme de control al versiunii - Git	8
2	Utilizarea platformei Gitlab	2
3	Sisteme de compilare - GNU Make şi CMake	4
4	Teste unitare - Google Test & Google Mock	2
5	Calculul ratei de acoperire a testelor - GCOVR	2
6	Generarea de documentaţie - Doxygen	2
7	Sisteme de containere - Docker	2
8	Cros-compilare folosind GCC/G++	2
9	Integrare continuă (CI) în Gitlab	4
	Total:	28

Bibliografie:

1. Documentaţia de referinţă Git - https://git-scm.com/docs
2. Scott Chacon şi Ben Straub, Pro Git, ediţia a II-a, editura Apress, 2024 - https://github.com/progit/progit2/releases/download/2.1.422/progit.pdf
3. Documentaţia Gitlab - https://docs.gitlab.com/
4. Documentaţia GNU Make - https://www.gnu.org/software/make/manual/make.pdf
5. Documentaţia CMake - https://cmake.org/cmake/help/latest/
6. Documentaţia Google Test - https://google.github.io/googletest/
7. gcovr, ghidul utilizatorului - https://gcovr.com/en/stable/guide.html
8. Manualul Doxygen - https://www.doxygen.nl/manual/index.html
9. Documentaţia Docker - https://docs.docker.com/get-started/overview/
10. Ruvinda Dhambarage, A master guide to Linux cross compiling - https://ruvi-d.medium.com/a-master-guide-to-linux-cross-compiling-b894bf909386
11. Documentaţie Gitlab CI - https://docs.gitlab.com/ee/ci/

LABORATOR

Nr. crt.	Conţinutul	Nr. ore

1	Implementarea sistemului de compilare	1
2	Dezvoltare proiect software	14
3	Scriere de teste unitare	7
4	Adnotare şi generare documentaţie	2
5	Crearea imaginii de Docker pentru compilare şi cros-compilare	2
6	Configurarea sistemului de integrare continuă în Gitlab	2
	Total:	28
Bibliografie:		

11. Evaluare

Tip activitate	11.1 Criterii de evaluare	11.2 Metode de evaluare	11.3 Pondere din nota finală
11.4 Curs	Însuşirea cunoştinţelor teoretice generale.	Test grilă pe platforma Moodle.	50%
11.5 Seminar/laborator/proiect	Studentul a înţeles şi a utilizat tehnicile în dezvoltarea proiectului.	Acumularea de puncte asociate sarcinilor de pe Gitlab, puncte acordate odată cu finalizarea cu succes a sarcinilor.	50%
11.6 Condiţii de promovare			
Studentul va obţine: • cel puţin un punct asociat cu o sarcină de tip ”dezvoltare”; • cel puţin un punct asociat cu o sarcină de tip ”testare”; • cel puţin un punct asociat cu o sarcină de tip ”proces”; • cel puţin un punct asociat cu o sarcină de tip ”documentaţie”; Studentul va obţine minim 50% din punctajul total.			

12. Coroborarea conţinutului disciplinei cu aşteptările reprezentanţilor angajatorilor şi asociaţiilor profesionale reprezentative din domeniul aferent programului, precum şi cu stadiul actual al cunoaşterii în domeniul ştiinţific abordat şi practicile în instituţii de învăţământ superior din Spaţiul European al Învăţământului Superior (SEÎS)

În contextul ingineriei software, companiile dezvoltă aplicaţii din ce în ce mai complexe, acestea având nevoie de un mediu de dezvoltare din ce în ce mai bine automatizat. Lipsa unui astfel de mediu duce la probleme grave de performanţă care măresc semnificativ timpul necesar implmentării complete a soluţiei. Astfel, disciplina este critică pentru orice inginer care lucrează în domeniu, pregătindu-l pe acesta cu un bagaj de cunoştinţe care să îi permită să fie eficient pe tot parcursul procesului de dezvoltare.

Universitatea Naţională de Ştiinţă şi Tehnologie Politehnica Bucureşti
Facultatea de Electronică, Telecomunicaţii şi
Tehnologia Informaţiei

Data completării	Titular de curs	Titular(i) de aplicaţii
21.09.2026	Conf. Dr. Radu Hobincu	Conf. Dr. Radu Hobincu

Data avizării în departament	Director de departament
22.09.2026	Prof. Dr. Claudius Dan

Data aprobării în Consiliul Facultăţii	Decan
22.09.2026	Prof. Dr. Mihnea Udrea